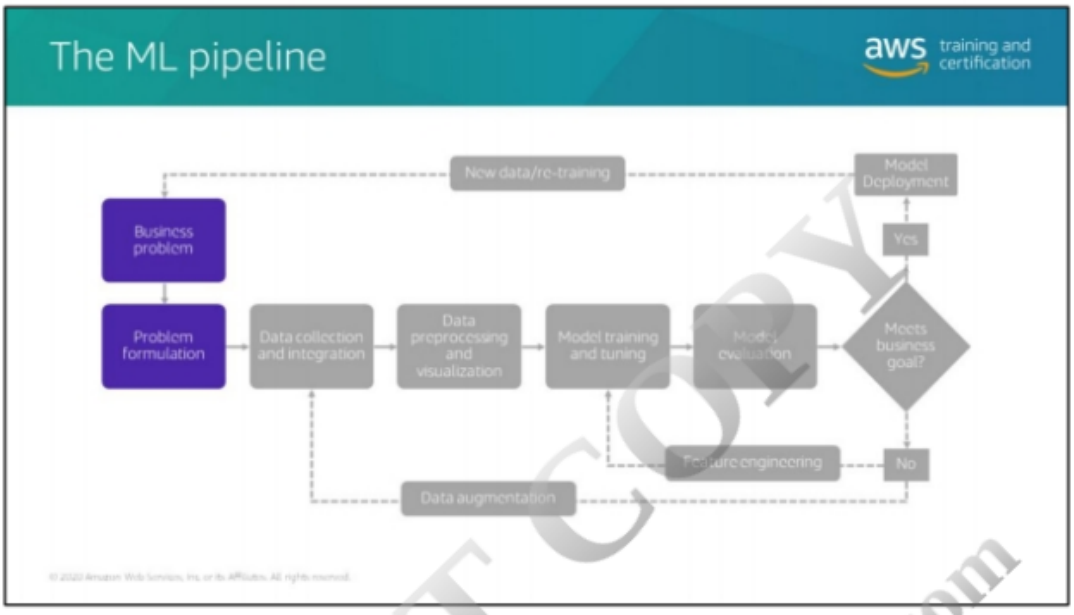
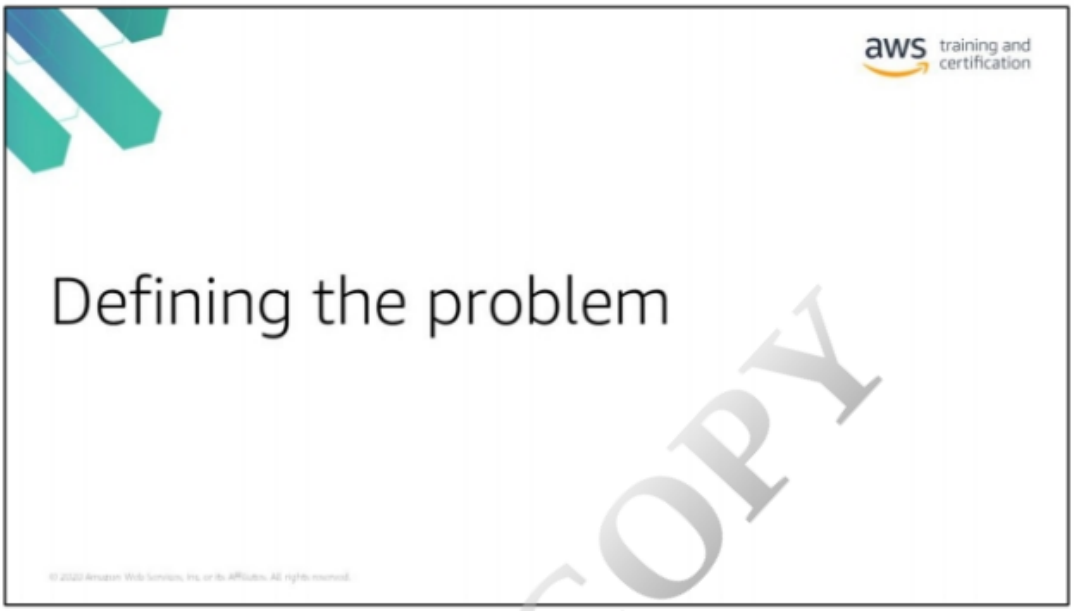
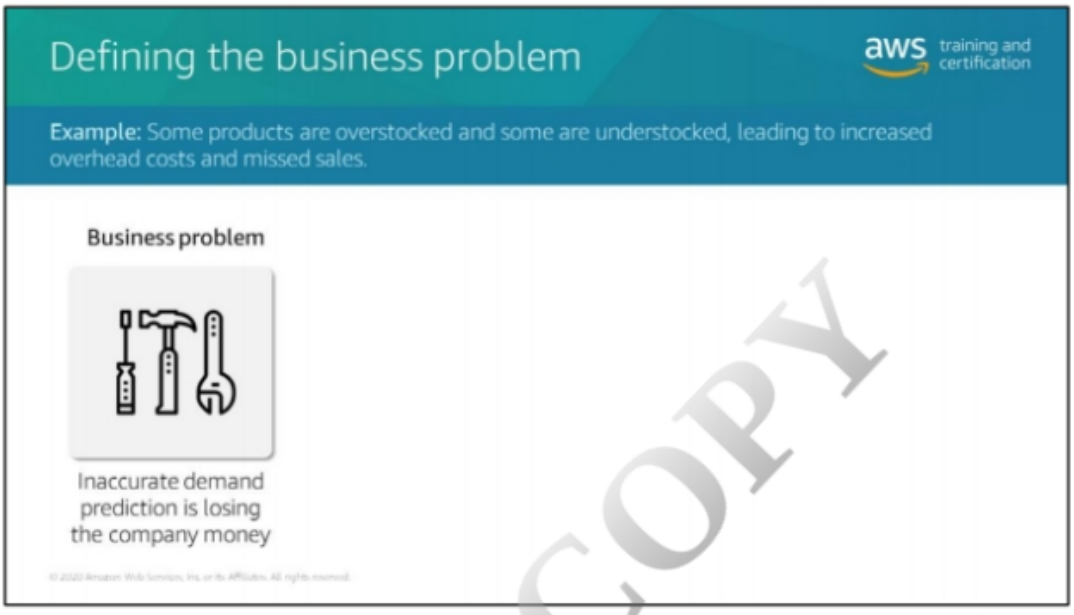


Before you can begin working in a machine learning pipeline, you need to have a business problem that can be turned into a machine learning problem. This sounds like a short, simple process – but it's actually fairly involved. Let's explore.

This module introduces you to the first phase of the ML pipeline: Problem formulation.

Problem formulation, a research-oriented phase of the pipeline, is the process of exploring and defining the problem you need to solve. It is the starting point for any ML project because you need a thoroughly defined problem in order to come up with an appropriate solution. This phase, and therefore this module, covers defining the problem, choosing data, and identifying success. Then we'll wrap up this module with some hands-on practice.

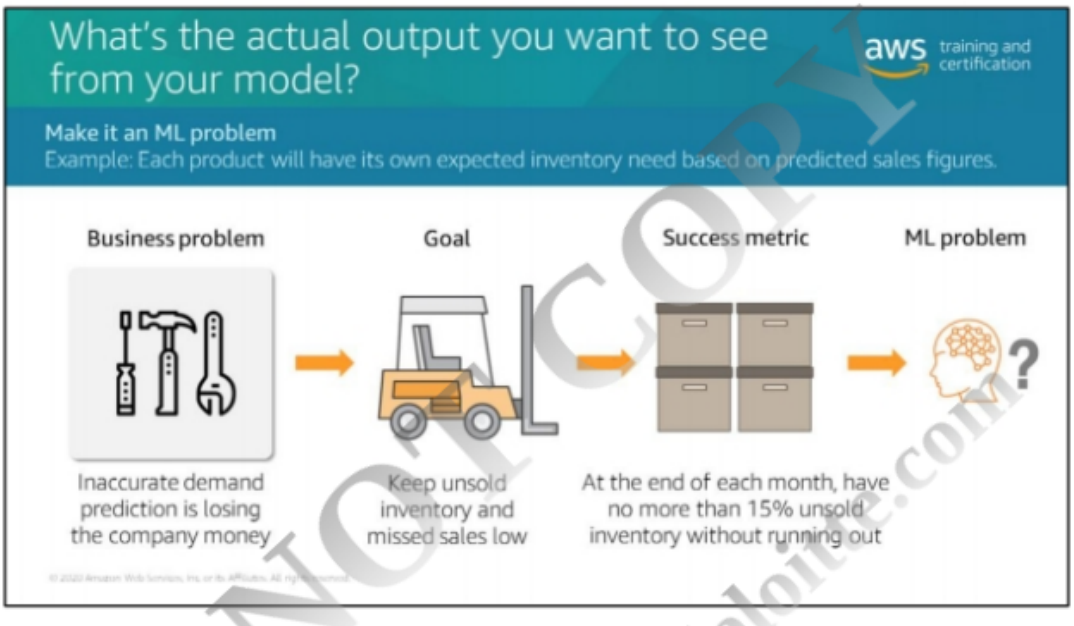




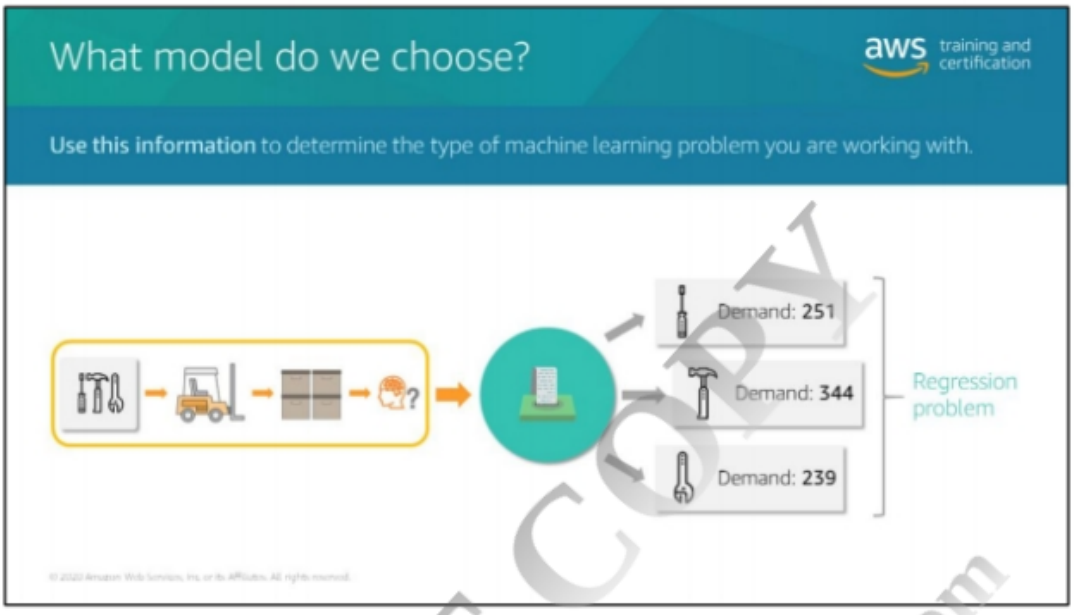
Your first step in this phase is to simply define the problem you're trying to solve and the goal you need to reach. Let's use the following example: we want to reduce the amount of unsold inventory left in stock.

Now, what's the business goal or outcome driving this problem statement? Reducing the amount of unsold inventory while not losing sales due to lack of inventory.

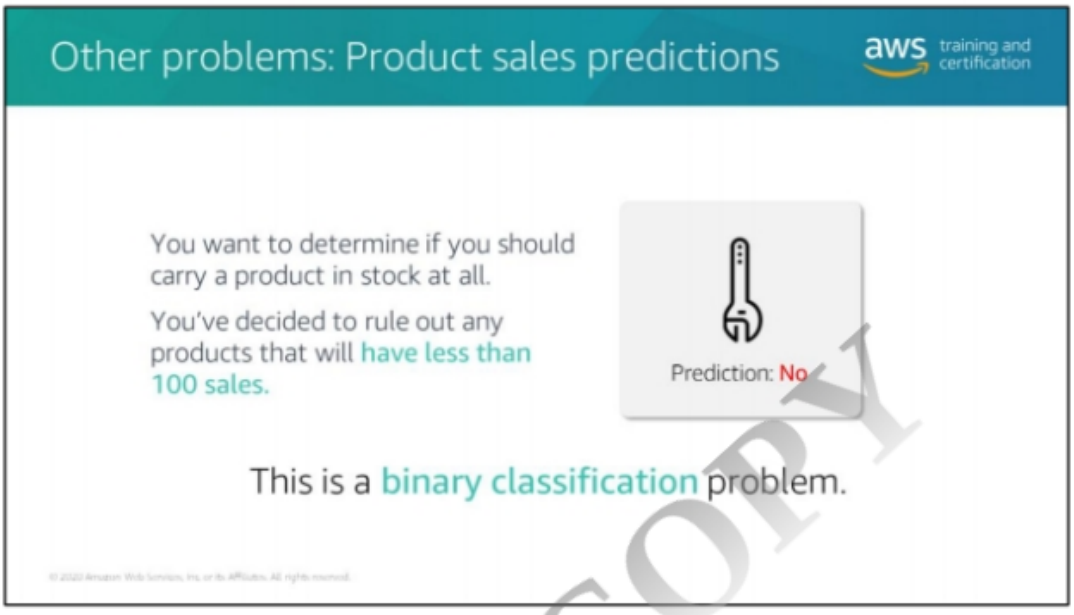
From a business perspective, how do you define success? This is the stage in which you must move from *qualitative* statements to *quantitative* statements that can be easily measured. Continuing with our example, a metric you could use to define success for this problem might be: At the end of the month, have no more than 15% unsold inventory without running out.



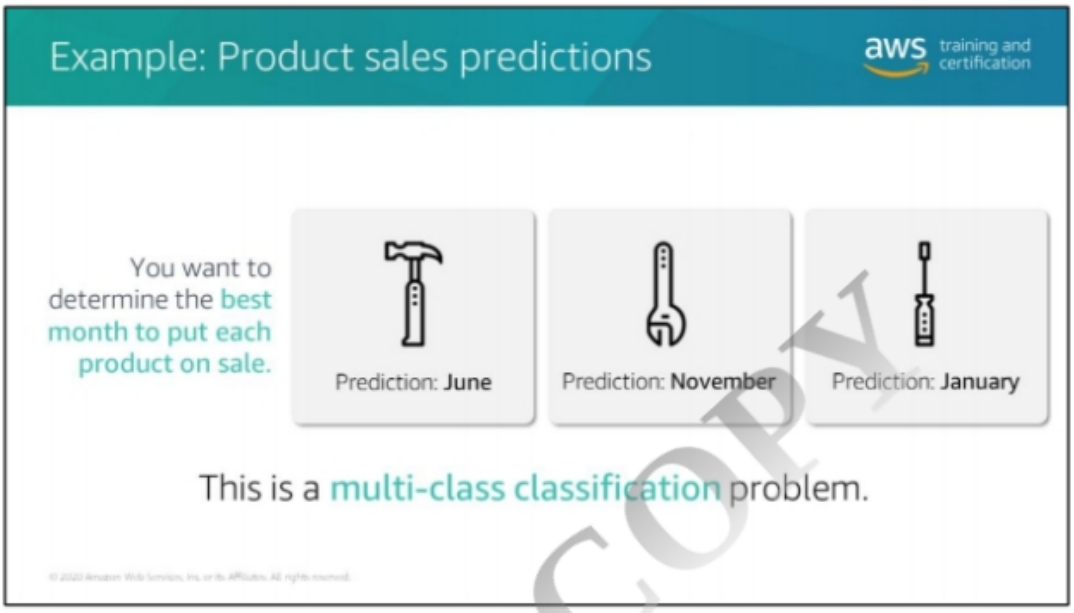
Now that you understand the business aspects of your problem, it's time to decide which ML model to use. What output do you want to see from your model? It pays to be specific; it should be a statement that reflects what a ML model could actually output.



Because you'll be predicting specific sales values for each item, this is most likely a regression problem.

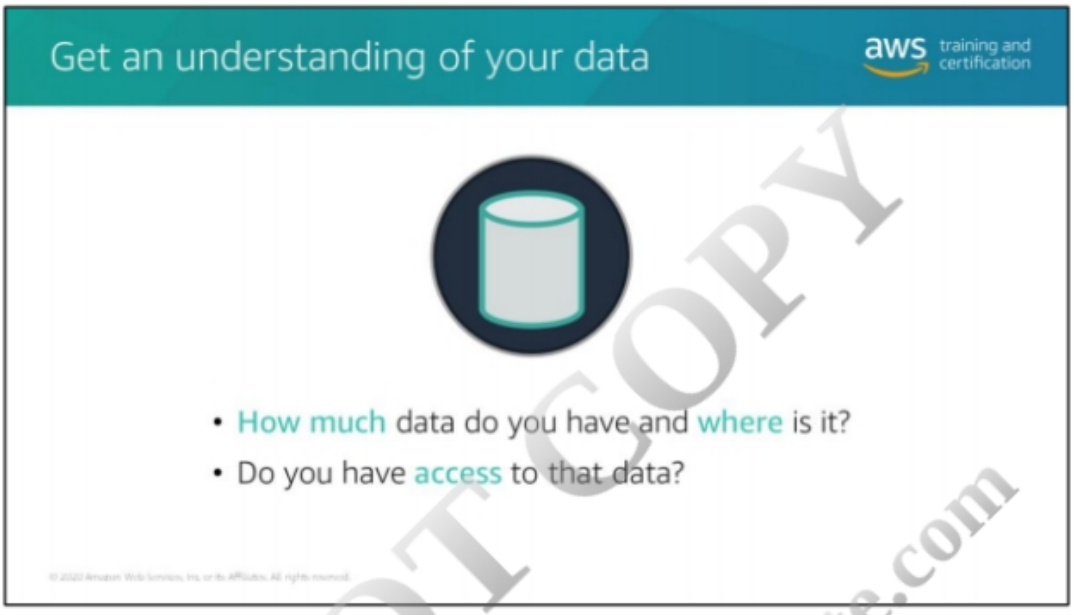


You could also have a different need that is a different kind of ML problem. For example, maybe you're trying to determine whether you should carry an item in stock at all.



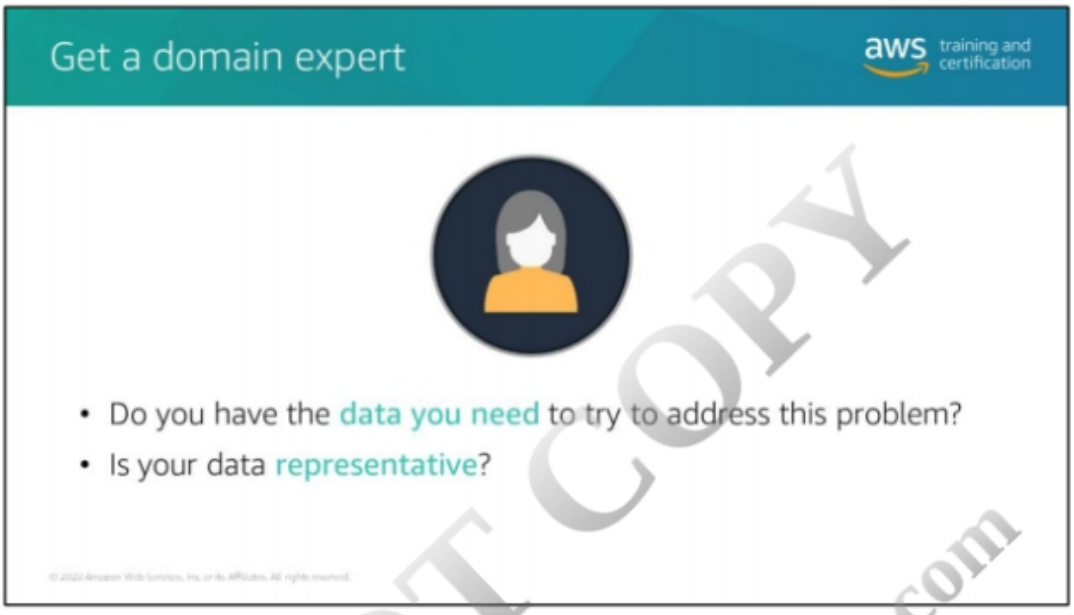
Or maybe you want to determine which month would be best to put each product on sale.

It is important to avoid overcomplicating the problem and to frame the simplest solution that meets your needs. However, it is also important to avoid losing information, especially information in the historical answers. Here, converting an actual past sales number into a binary variable like "more than 10" would lose valuable information. Investing time in deciding which target makes most sense for you to predict will save you from building models that don't answer your question.



Let's get back to our original example about predicting credit card fraud. We've further formulated the problem, but what data do we need to actually train our model to reach the desired output and subsequently our intended business outcome? Do we have access to that data? If so, how much data do you have and where is it? What solution can we use to bring all of this data into one centralized repository? These questions are essential to answer at this stage.

Before answering these questions, however, let's take a step back and talk quickly about the typical elements making up a ML dataset.



So, given what we know about the elements of a ML dataset, let's get back to one of our original questions: what data do we need to actually train our model to reach the desired output, and subsequently our intended business outcome? This is an example of a stage within the ML pipeline when it's vital to get domain expertise to help you answer this question. With domain knowledge, you can begin to determine the features and target data your model will need to make accurate predictions.

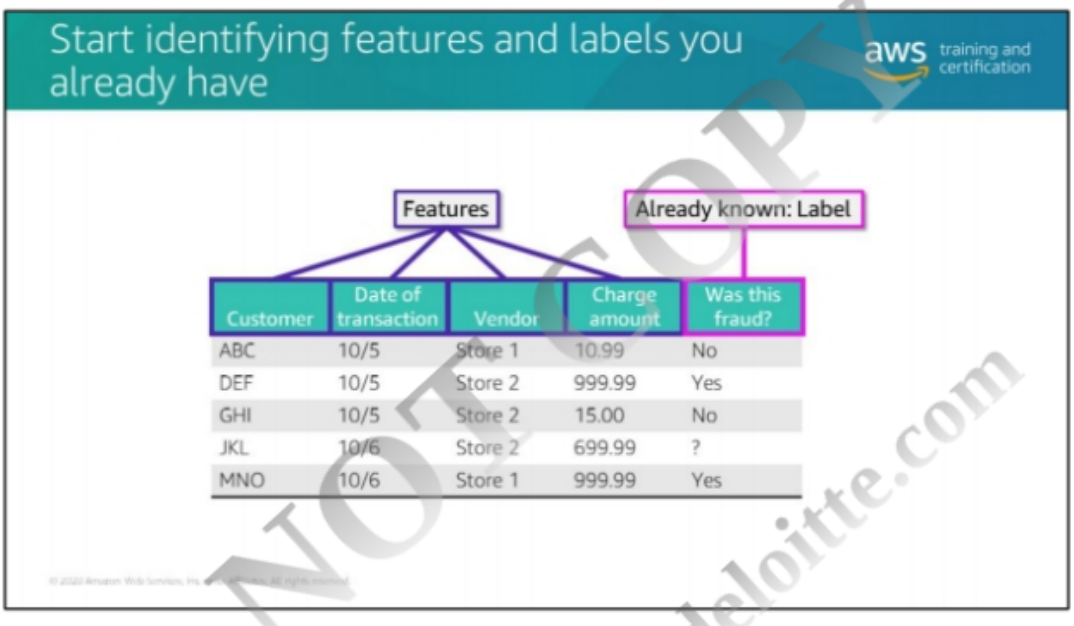
Your data should be representative of the data that you will have when you are using the model to make a prediction. For example, if we want to predict credit card fraud, we must collect both positive (fraudulent transactions) and negative (non-fraudulent transactions) data for the machine learning algorithm to be able to find patterns that will distinguish between the two types. If your average amount of fraudulent transactions is actually 3%, but your training dataset only includes a very small fraction of fraudulent observations, say 0.4%, it will be hard for your model to truly learn patterns related to fraudulent transactions it might encounter in production.

Evaluate the quality of your data too

aws training and certification

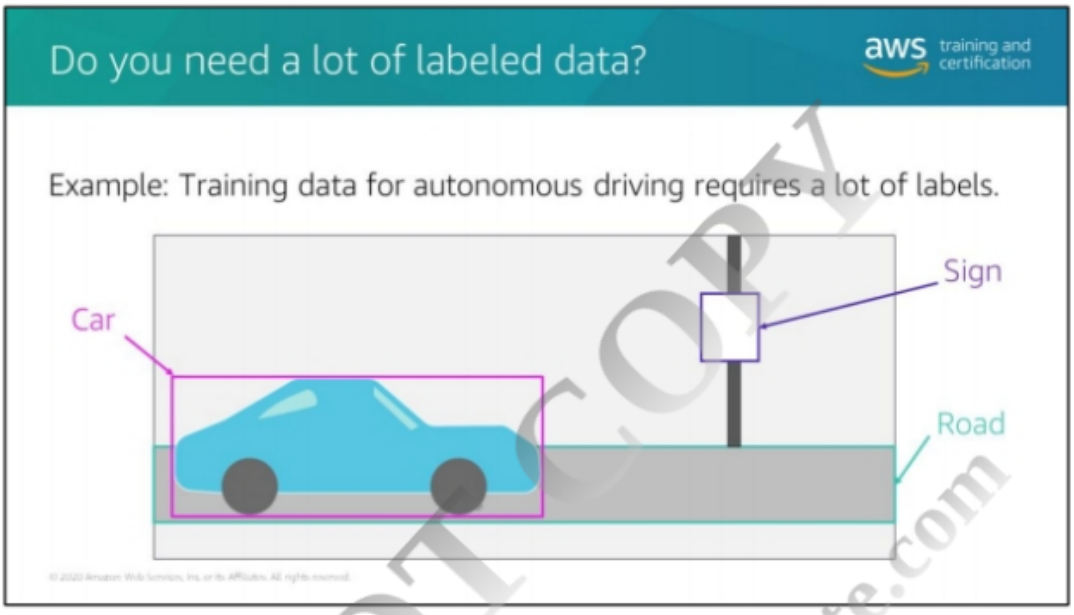
Product name	Price	Max stock	Current stock	Sales this week
Soap	1.99	20	14	49
Shampoo	6.99	20	2	23
Hair brushes	12.95	30	12	2
Toothpaste	3.50	30	13	40
Toothbrushes	5.00	20	?	?
Lotion	8.75	10	?	?

A general rule of thumb here is that good data will contain a signal about the phenomena that you're trying to model. For instance, let's say a wholesaler is trying to forecast demand for a few of their products. They might track the number of sales they've had previously, which is a good start—but what if they forget to log when certain products were out of stock? If you're trying to forecast demand, it's important to know when you were out of stock and, therefore, critical to have that data represent one of your features.

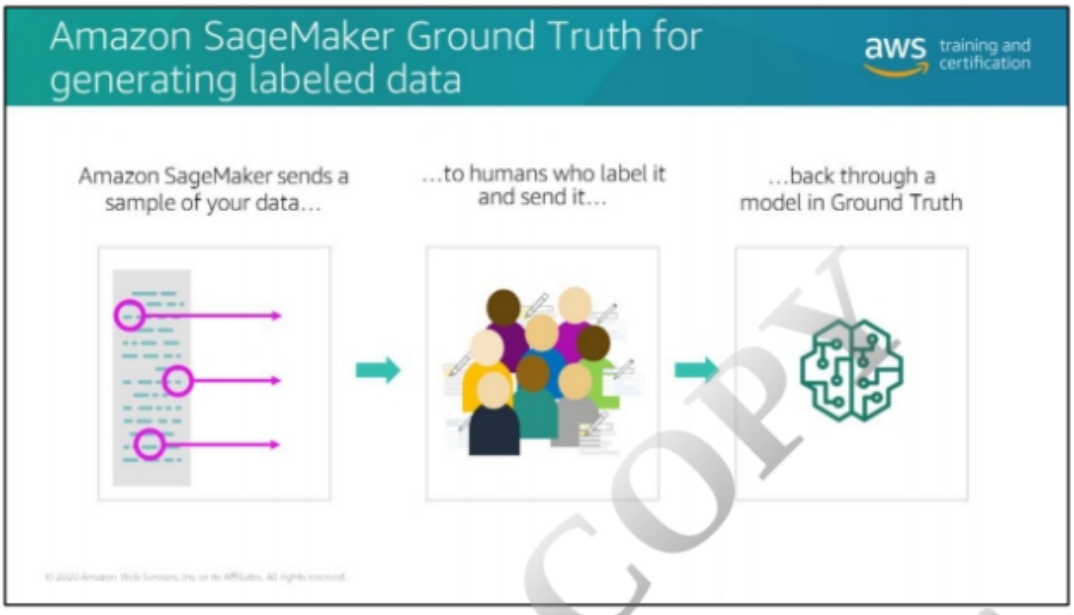


Your data is made up of two elements: features and labels. A *feature* is an attribute that can be used to help identify patterns and predict future answers. A feature in our credit card example could be the "date of transaction," the "vendor," and/or the "amount, in dollars, of the transaction."

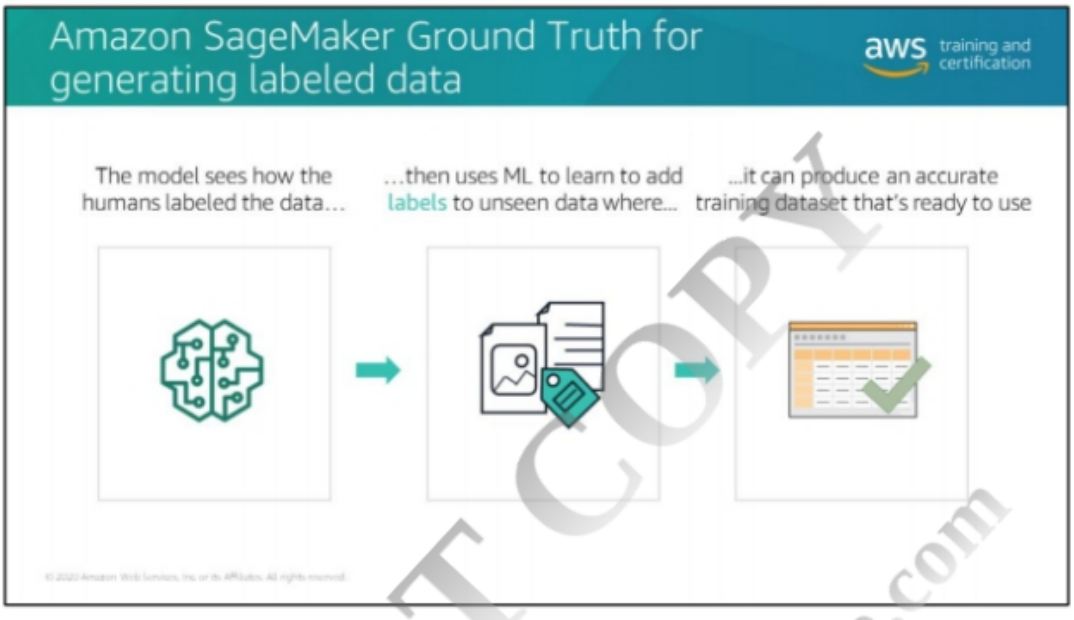
Subsequently, data for which you already know the answer is called *labeled* data. The label is the answer that you want your model to predict. So in our credit card transaction example, the label of any given observation is either "fraud" or "not fraud."



Many of the examples we've been discussing so far represent supervised learning and therefore include labeled data. But what if you don't have labeled data? In some cases, getting the labeled data that you need can be very arduous. Take the use case of autonomous driving. Image data needs to be precisely labeled with where exactly the car is at any given moment and where it might be relative to some other object like the road or a building or a road sign. Without this labeling, the trained model would not be as accurate, and could lead to potentially fatal mistakes.



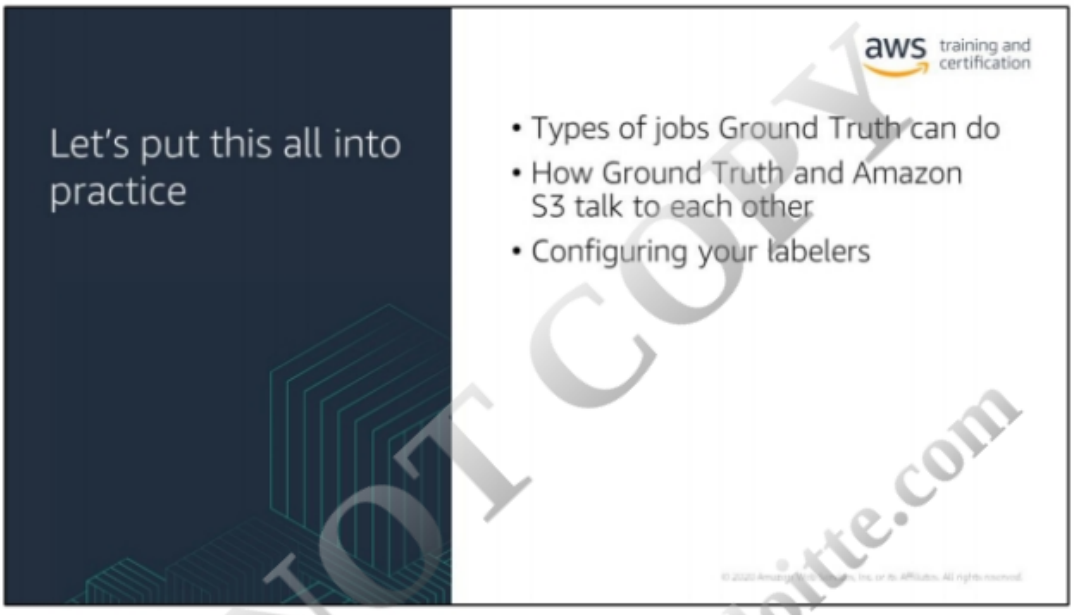
With Amazon SageMaker Ground Truth, you can use a combination of a selected human workforce and machine learning to create a labeled dataset. Ground Truth gives you access to teams of human labelers that label your data, and then feed that labeled data back to the service for automated labeling.



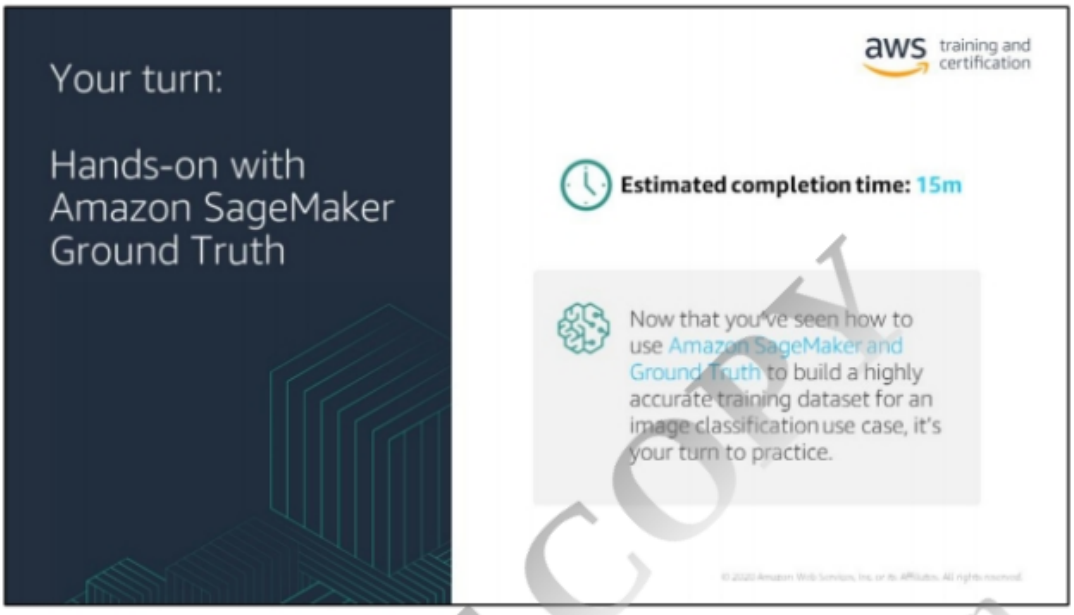
Ground Truth uses machine learning to learn how to label the data like the humans did, then conducts automated labeling on unseen data, and finally produces a dataset you can use for training.

Use the workforce of your choice to label your dataset. You can choose your workforce from:

- The Amazon Mechanical Turk workforce of more than 500,000 independent contractors worldwide. Think of this as a public crowd that is the most available and cost effective for your labeling tasks. This is most likely the best option if your data is publicly accessible and not confidential.
- However, if your data is confidential, you can use a private workforce that you create from your employees or contractors for handling data within your organization.
- A third option, which may also work for your confidential data, is to use a vendor that you can find in the AWS Marketplace that specializes in data labeling services with their NDA's and SLA's. This may be a cheaper option than using your employees.

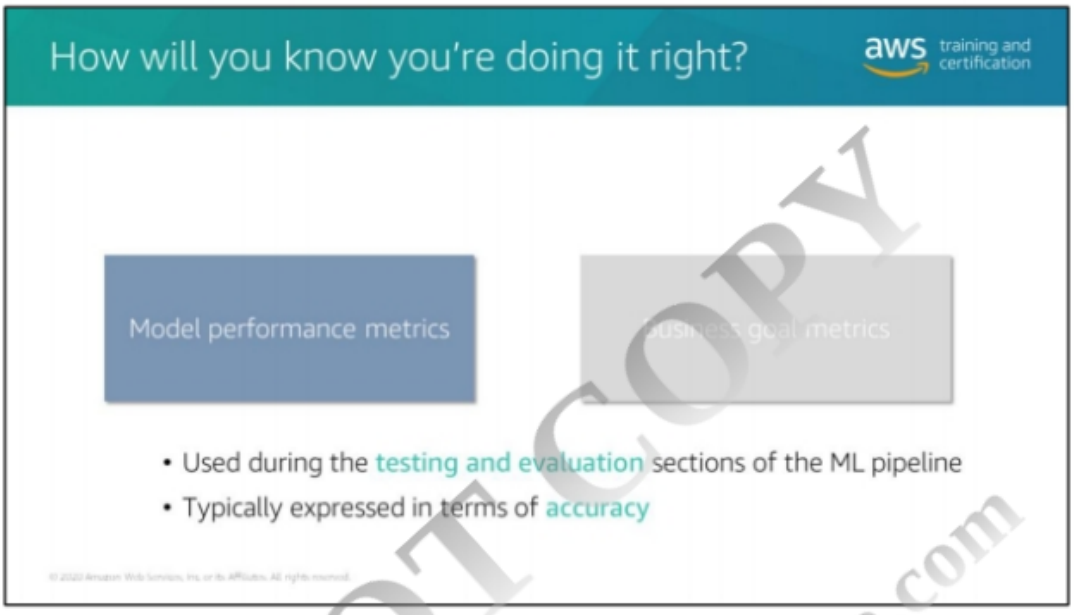


To get started with Amazon SageMaker Ground Truth, let's dive into the service. We'll start a labeling job so that we can see how Ground Truth saves us time and resources, we'll see how Ground Truth and Amazon S3 work together to store our labeled dataset, and then we'll configure our labelers. As we move through this demo, consider *when* and *how* Ground Truth may be a value-add for your machine learning business use cases.

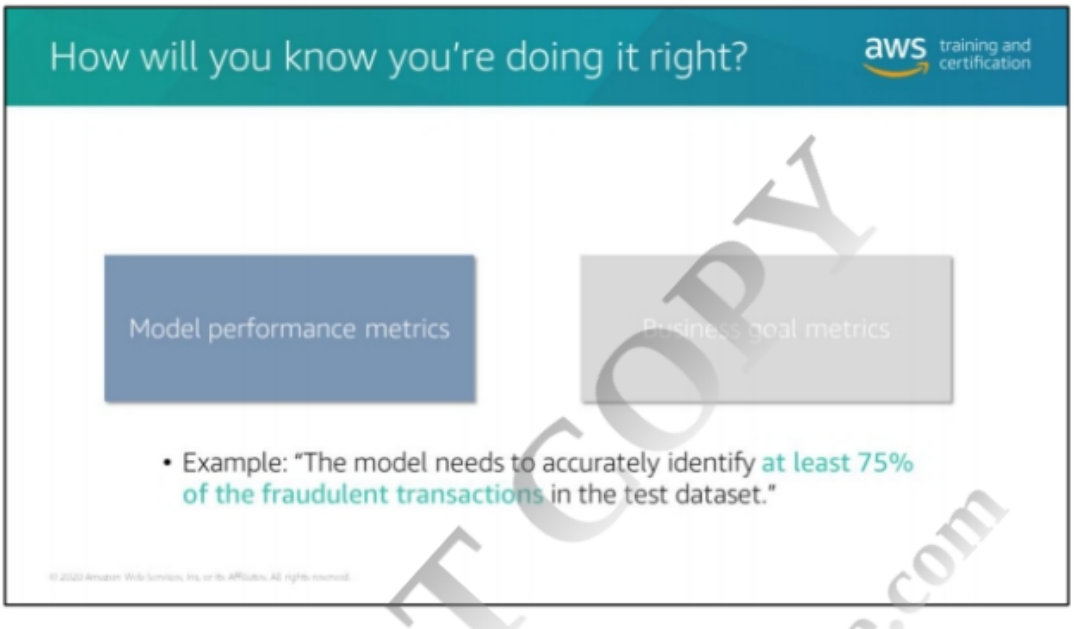


You'll now have an opportunity to gain hands-on experience with Amazon SageMaker Ground Truth by completing a tutorial of the service. You'll have 15 minutes to complete the walk-thru. https://aws.amazon.com/getting-started/tutorials/build-training-datasets-amazon-sagemaker-ground-truth/

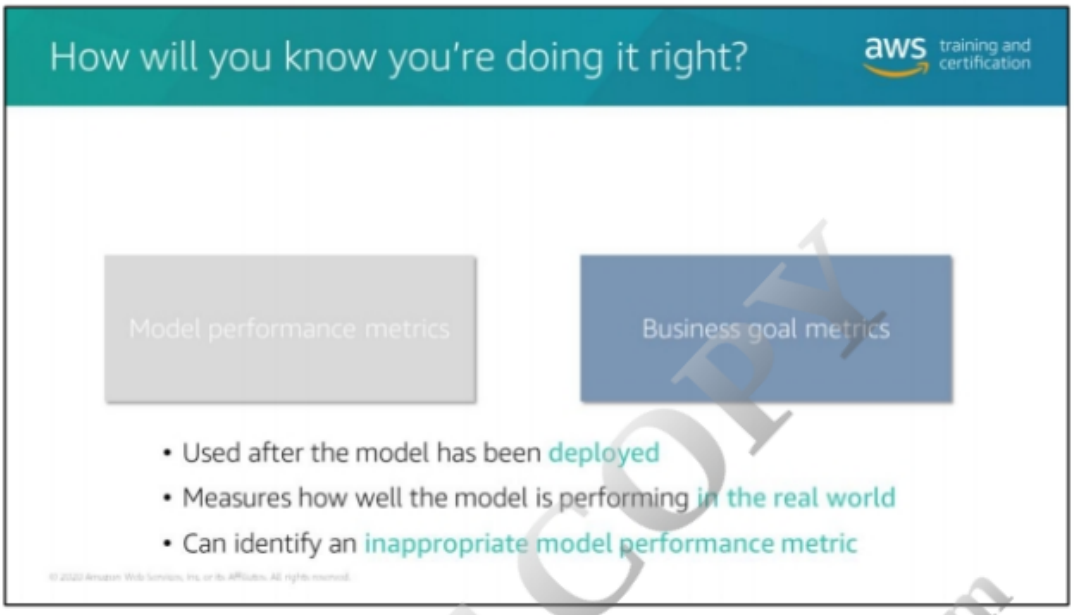
So you've defined your problem and metrics for successfully solving it, but how do you know what you're actually achieving?



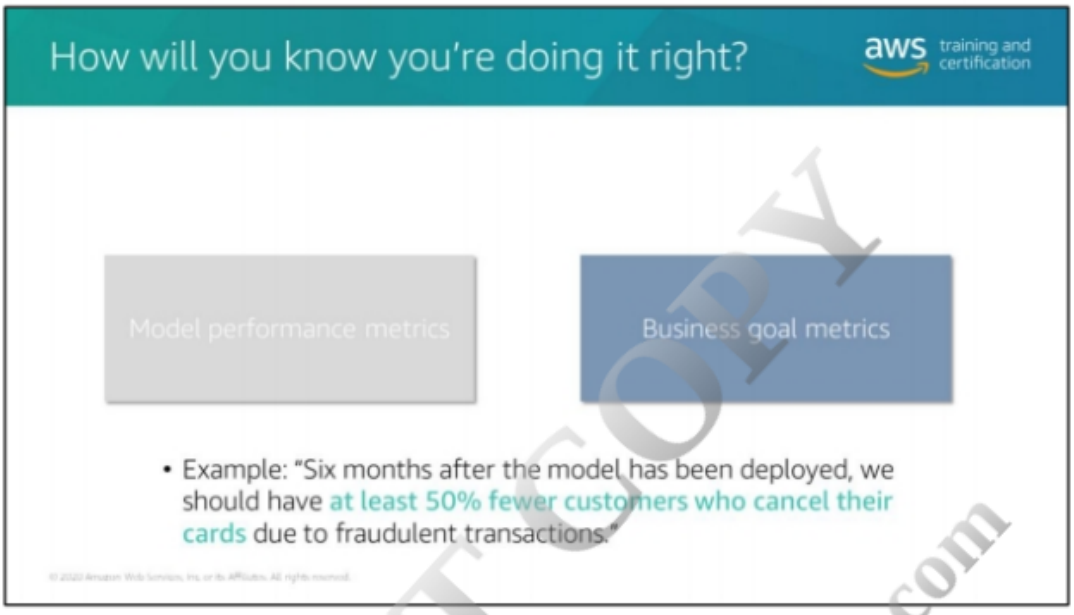
You'll keep an eye on how your model's actually performing during the testing and evaluation phases. This type of information is typically judged by how accurate the model is.



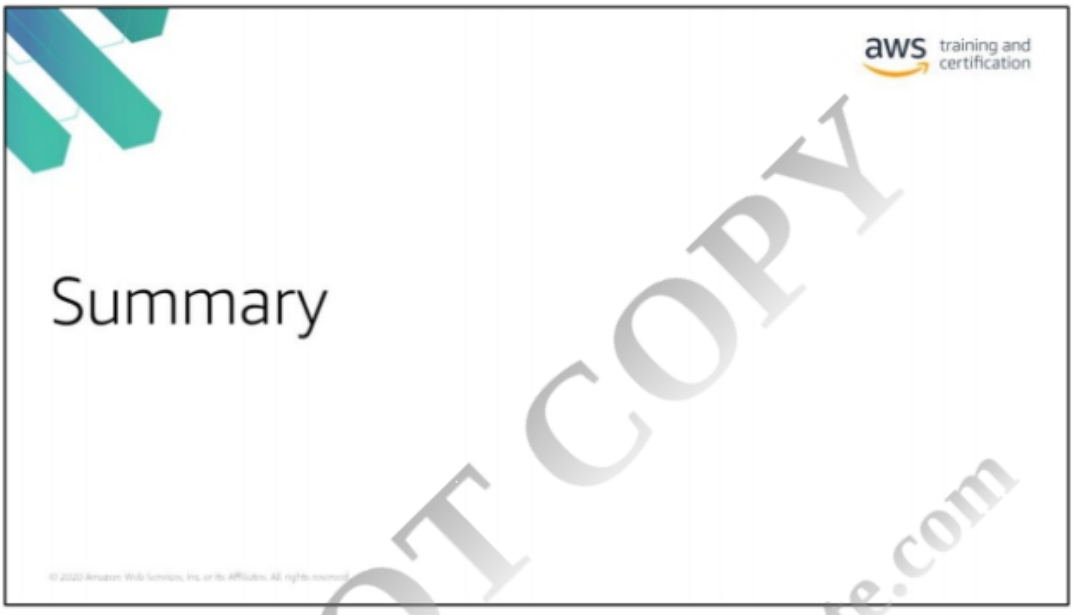
An example could be that you're looking to see if the model is accurately identifying at least 75% of the fraudulent transactions in the test dataset.

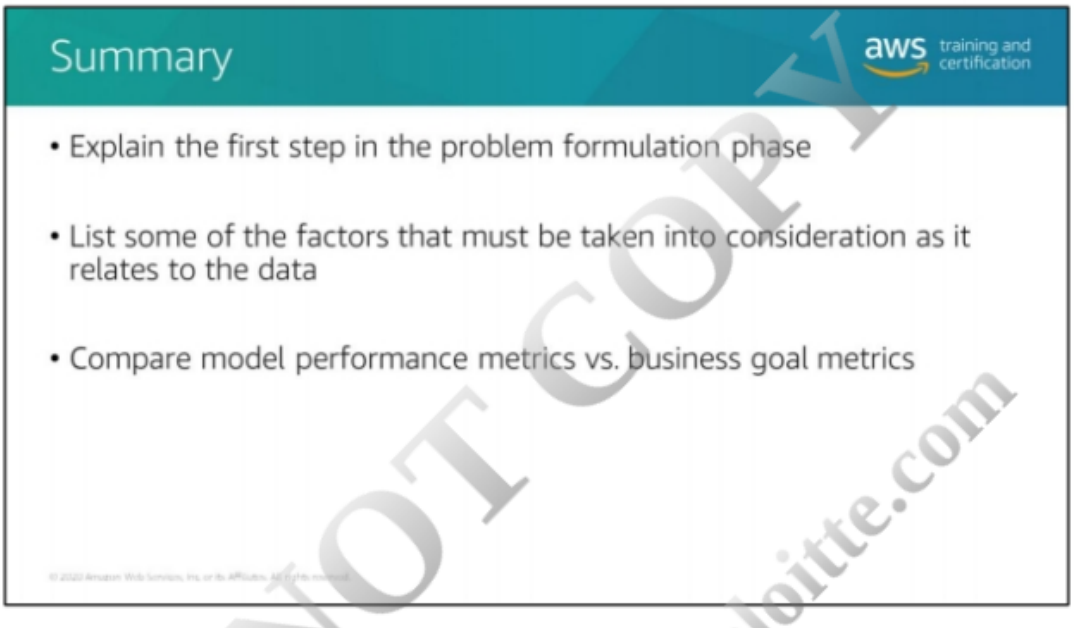


Whereas you'll keep an eye on whether or not your business goal is being met after the model has been deployed.



An example could be that you look to see if after 6 months in deployment, do you have at least 50% fewer customers who cancel their cards due to fraudulent transactions.





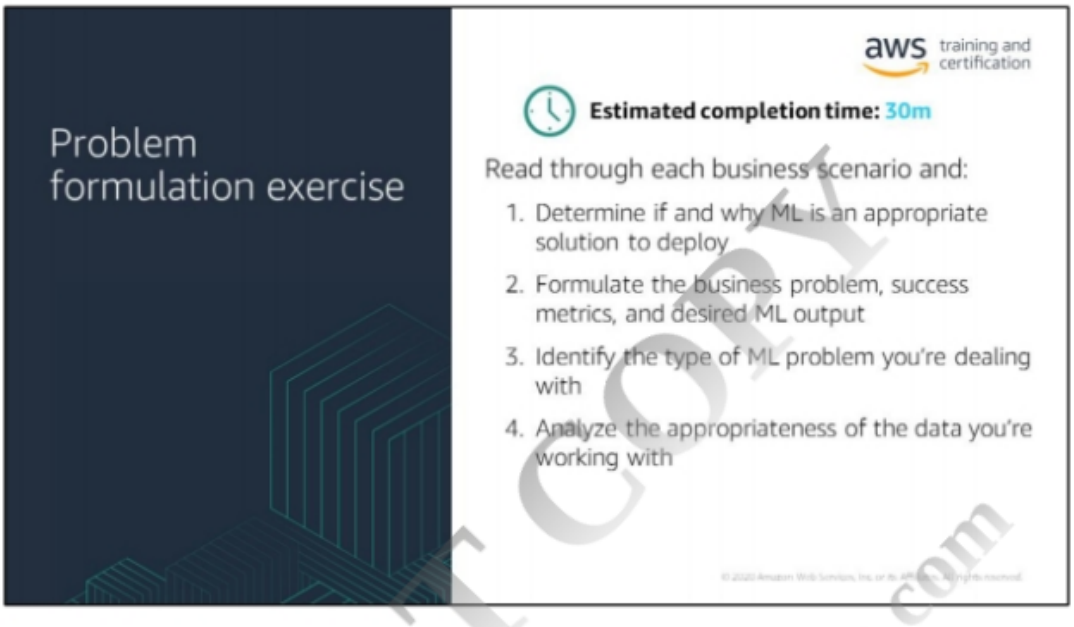
We've covered a lot in module one. You should now be able to:

Explain the first step in the problem formulation phase
- Define the problem you're trying to solve and the goal you need to reach.

List some of the factors that must be taken into consideration as it relates to the data
- Amount of data
- Where the data is and is it accessible
- Do you have access to the data?
- Is the data representative?
- What is the quality of the data?

Compare model performance metrics vs. business goal metrics
Model performance metrics:
- Used during the testing and evaluation sections of the ML pipeline
- Typically expressed in terms of accuracy

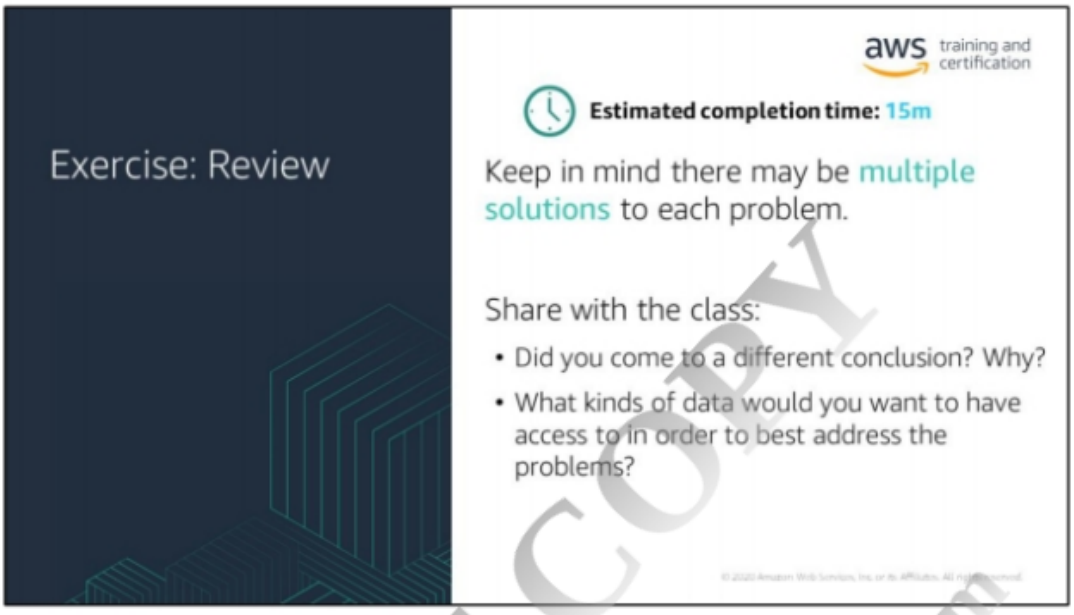
Business goal metrics
- Used after the model has been deployed
- Measures how well the model is performing in the real world
- Can identify an inappropriate model performance metric



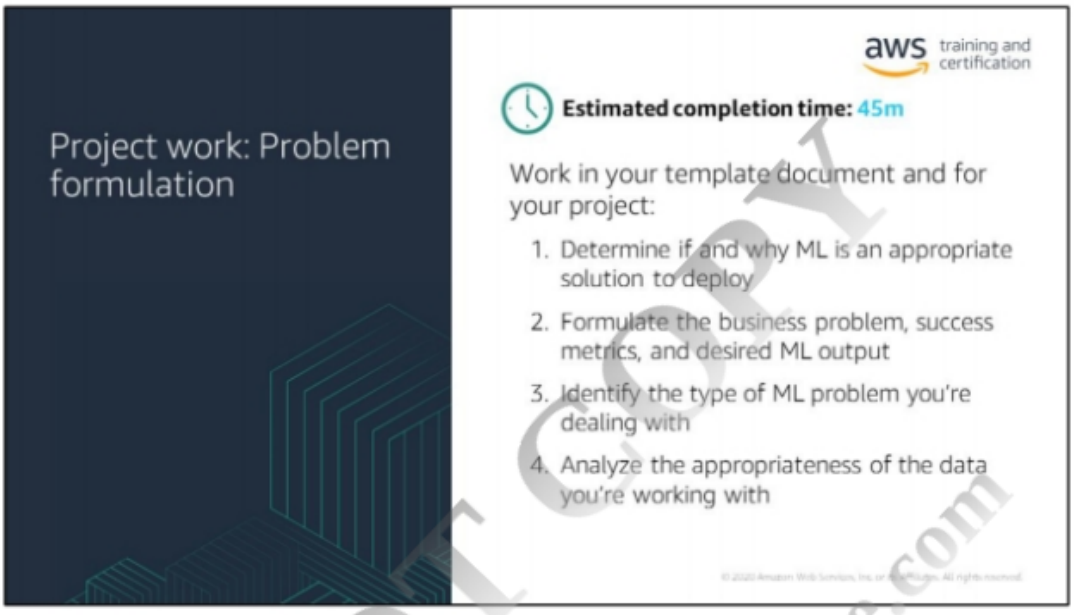
Over the next 30 minutes, you'll get the chance to apply some of the concepts we just discussed related to the problem formulation phase of the ML pipeline to a hands-on exercise. More specifically, in this exercise, you'll get be asked to read through a business scenario and:

1. Determine if and why ML is an appropriate solution to deploy
2. Formulate the business problem, success metrics, and desired ML output
3. Identify the type of ML problem you're dealing with
4. Analyze the appropriateness of the data you're working with
5. Start to think about the algorithms you might want to use for the problem

Accessing the Problem Formulation exercise – To get started with this exercise, please visit: https://aws-tc-largeobjects.s3-us-west-2.amazonaws.com/ILT-TF-200-MLDWTS/Problem+Formulation+Exercise.docx



Solutions can be found here https://aws-tc-largeobjects.s3-us-west-2.amazonaws.com/ILT-TF-200-MLDWTS/Problem+Formulation+Exercise+Solutions.docx



Refer to the project template that was provided, or you can create your own.

Project template here: https://aws-tc-largeobjects.s3-us-west-2.amazonaws.com/ILT-TF-200-MLDWTS/Student+Project+Template.docx